TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

DÁLKOVÉ OVLÁDÁNÍ PRŮZKUMNÉHO ROBOTA

Bakalářská práce

Studijní program: B2646 – Informační technologie
Studijní obor: 1802R007 – Informační technologie

Autor práce: **Miroslav Klouda**
Vedoucí práce: Ing. Zbyněk Mader Ph.D.

Liberec 2014

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

REMOTE CONTROL ROBOT EXPLORATION

Bachelor thesis

Study programme: B2646 – Informational Technologies
Study branch: 1802R007 – Informational Technologies

Author: **Miroslav Klouda**
Supervisor: Ing. Zbyněk Mader Ph.D.

Liberec 2014

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: 2013/2014

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Miroslav Klouda**
Osobní číslo:	**M11000094**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Dálkové ovládání průzkumného robota**
Zadávající katedra:	**Ústav informačních technologií a elektroniky**

Zásady pro vypracování:

1. Navrhněte modul dálkového ovládání průzkumného robota založeného na ovládacích prvcích gamepadu (tlačítka a joystiky).
2. Návrh zrealizujte na mikroprocesorech PIC32, vytvořte vhodnou koncepci přenosu dat RF signálem mezi dálkovým ovladačem a robotem.
3. Dálkový ovladač bude vybaven grafickým LCD displejem zobrazující směr pohybu průzkumného robota.

Rozsah grafických prací:	**Dle potřeby dokumentace**
Rozsah pracovní zprávy:	**cca 30 stran**
Forma zpracování bakalářské práce:	**tištěná/elektronická**

Seznam odborné literatury:

[1] **Lucio Di Jasio, Programming 32-bit Microcontrollers in C - Exploring the PIC32, 2008, ISBN: 978-0-7506-8709-6**

[2] **David Money Harris, Sarah L. Harris, Digital Design and Computer Architecture, 2013, ISBN: 978-0-12-394424-5**

Vedoucí bakalářské práce:	**Ing. Zbyněk Mader, Ph.D.** Ústav informačních technologií a elektroniky
Datum zadání bakalářské práce:	**12. září 2013**
Termín odevzdání bakalářské práce:	**16. května 2014**

V. Kopecký

prof. Ing. Václav Kopecký, CSc.
děkan

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
vedoucí ústavu

V Liberci dne 12. září 2013

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Datum:

Podpis:

Abstrakt

Bakalářská práce se zabývá návrhem a realizací dálkového ovládání pro průzkumného robota. Ovladač je navržen tak, že komunikace mezi uživatelem a řídícím mikroprocesorem probíhá pomocí dvou joysticků a LCD displejem. Komunikace robota a řídícího mikroprocesoru je uskutečněna přes radiové vlny. Dálkové ovládání je využitelné pro různé typy mechanických robotů od pojízdných robotů po létající quadricoptery.

První část práce obsahuje popis řídícího prvku dálkového ovladače, produkt společnosti Digilent Inc., mikroprocesor PIC32MX340F512H, bez kterého by dálkové ovládání nemohlo být zrealizováno. Poté následuje popis modulu Pmod Schield-Uno, který umožňuje přístup periferních modulů. Pomocí něj je připojen PmodRF2 s jeho bezdrátovým přijímačem/vysílačem MRF24J40. Dalším hardwarovým prvkem, který je v práci popsán jsou joysticky, jejich princip a způsob využití na tomto ovladači. Práce také obsahuje popis LCD displeje od firmy HAOYU Electronics, který zde zastupuje komunikaci od mikroprocesoru k uživateli. Jsou zde uvedeny jeho vlastnosti a základní charakteristika ovladače SPFD5408.

V hlavní části se práce zabývá popisem návrhu a zapojení mikroprocesoru s dálkovým ovládáním, tzn. připojení joysticků a LCD displeje. Dále práce pojednává o vytvoření komunikace přes SPI a bezdrátový přenos dat modulem PmodRF2. V této části práce je uveden algoritmus pro načtení dat z joysticků, zobrazování dat na displeji a odeslání příkazů robotovi.

Klíčová slova

dálkové ovládání, pic32, mikroprocesor, bezdrátový přenos, LCD, SPFD5408, SPI, MRF24J40

Abstract

The bachelor thesis deals with the design and implementation of remote control for exploration robot. The driver is designed so that communication between the user and the microprocessor is realised via two joysticks and an LCD display. The robot and the microprocessor communicate through radio waves. The remote control is usable for different types of mechanical robots from the mobile ones to the flying quadricopters.

The first part contains a description of the remote control driver, Digilent Inc. company product, microprocessor PIC32MX340F512H, without which the remote control can't be implemented. This part is followed by a description of the Pmod Schield-Uno module which allows access to peripheral module and which also enables the connection of PmodRF2 with its wireless receiver / transmitter MRF24J40. Other hardware elements described in this thesis are joysticks, their principle and methods of their usage. The thesis also contains a description of the LCD display from HAOYU Electronics representing the communication from the microprocessor to the user. The description presents the properties and basic characteristics of drivers SPFD5408.

The main part of the thesis deals with the description of the design and the connection between the microprocessor and the remote control i.e. joysticks and LCD display connection. Further on the thesis deals with the creation of communication via SPI and wireless data transmission by PmodRF2 module. In this part of the thesis an algorithm to get data from the joystick, show data on the display and send commands to the robot is specified.

Key words

remote control, pic32, microprocesor, wireless transmission, LCD, SPFD5408, SPI, MRF24J40

Poděkování

Tímto chci poděkovat panu Ing. Zbyňku Maderovi, Ph.D. za vstřícný přístup při vedení této bakalářské práce. Děkuji za jeho rady při sestavování dálkového ovládání, užitečné konzultace ohledně této práce, ochotu pomoci nalézt optimální řešení a další aktivity spojené s dokončením této práce.

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

SW Software

HW Hardware

RAM Random-Access Memory

LCD Liquid Crystal Display

LED Light-Emitting Diode

OLED Organic Light-Emitting Diode

PWM Pulse Width Modulation - Pulzně šířková modulace

IDE Integrated Development Environment

MPIDE Multi Platform Integrated Development Environment

MIPS Million Instruction Per Second

SPI Serial Peripheral Interface

IO Input/Output

PWM Pulse-Width Modulation

UART Universal Asynchronous Receiver/Transmitter

ADC Analog to Digital Converter

WDT Watch Dog Timer

RF Radio Frequency

RGB model Red, Green, Blue

SoC System on Chip

CSMA-CA Carrier Sense Multiple Access with Collision Avoidance

MOSI Master Out, Slave In

MISO Master In, Slave Out

CLK Clock

CS Chip Select

CRC Cyclic redundancy check - Kontrolní součet

PLL Phase-Locked Loop = Fázový závěs

RSSI Received signal strength indicator = Indikátor intenzity signálu

Úvod

V dnešní době si bez elektroniky téměř neumíme představit náš každodenní život. Pomáhá nám v běžných činnostech, v zaměstnání, doma, při sportu, atd. Díky ní můžeme ovládat téměř jakýkoliv produkt, např. garážová vrata, domácí spotřebiče, automobily a mnoho dalších často používaných zařízení.

Abychom měli tuto možnost, je potřeba, aby v daném produktu byla řídící jednotka, mikroprocesor. Ten jej bude řídit na základě předem daných parametrů. Z tohoto důvodu je vývoj mikroprocesorů velice důležitý ať už se jedná o jejich velikost, paměť nebo rychlost. Cílem vývoje je optimalizovat tyto procesory pro účely daných produktů. Samotné mikroprocesory by ovšem nestačily. Důležitým prvkem mikroprocesoru je software, na základě kterého vykonává příkazy určující jeho činnost.

Cílem této bakalářské práce je vytvoření dálkového ovládání pro průzkumného robota. To zahrnuje sestavení jednotlivých komponent tak, aby vytvořily prototyp dálkového ovládání a dále je úkolem navrhnout software pro tento prototyp. Software bude obsahovat zdrojové kódy pro zobrazování dat na displeji, čtení dat z joysticků a odesílání dat robotu. Tedy ovladače, jehož úkolem je bezdrátově řídit všesměrového pojízdného robota.

Součástí dálkového ovládání budou dva analogové joysticky, barevný LCD displej a RF modul. Joysticky určují tyto ukazatele: rychlost a směr. LCD displej zobrazuje data (rychlost, směr a otočení) uživateli, podle kterých uživatel získává informace o pohybu robota. RF modul slouží pro komunikaci mikroprocesoru s robotem.

Řídící jednotkou, která je v práci použita je mikroprocesor PIC32MX340F512H. Tato jednotka bude naprogramována v jazyce C a bude řídit příkazy zasílané na displej v závislosti na datech od uživatele. Tyto příkazy jsou dány pohyby joysticků. Další činností řídící jednotky je odesílání dat přes RF modul robotu s příkazy určujícími jeho pohyb.

Splnění zadání této práce znamená vytvoření funkčního prototypu univerzálního dálkového ovládání pro pohyblivého robota, ale v této bakalářské práci je systém vyvinut pro konkrétní typ. Pokud bude software robota schopen dekódovat přijaté příkazy, lze toto dálkové ovládání využít v různých oblastech běžného života.

1. Všesměrový pohyblivý robot

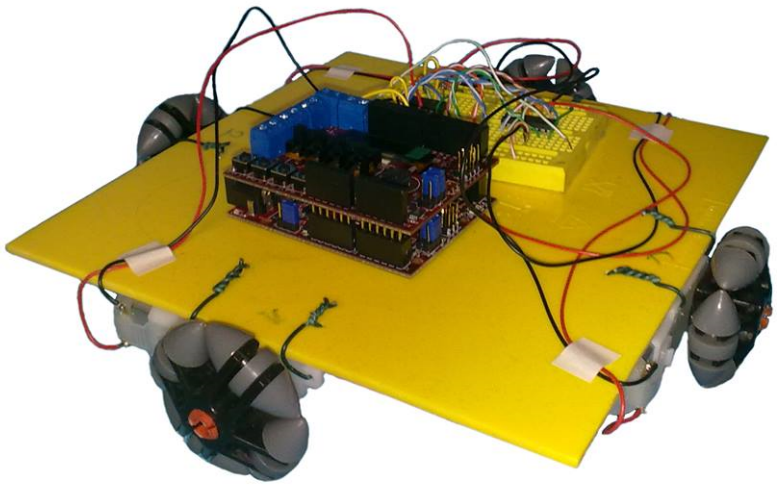
Dálkové ovládání, o kterém pojednává tato bakalářská práce, je vytvořeno pro všesměrového robota (1.1). Ten byl vytvořen v rámci ročníkového projektu na této fakultě.

Tento robot obsahuje stejný typ mikročipu a vývojového kitu, jaký je použit v dálkovém ovládání. Vývojový kit robota je ještě rozšířen o modul chipKIT Basic I/O Shield™.

Basic I/O Shield obsahuje OLED displej, 8 LED diod, 4 tlačítka, 4 přepínače a EEPROM paměť. LCD displej na tomto modulu je využit k zobrazování rychlosti ve směru osy X, osy Y a také směru pohybu.

Pro řízení pohybu robota jsou využity 4 motory, které řídí rychlost a směr otáčení koleček. Na motor je tedy zapsána hodnota pro směr otáčení motoru a dále je zapsán PWM signál, který nabývá hodnot v rozmezí od 0 do 100. Tyto hodnoty vyjadřují procentuální rychlost v daném směru otáčení.

Po dokončení tohoto dálkového ovládání bude nutné do SW robota přidat příkazy pro čtení a správné dekódování přijatých dat z RF modulu.

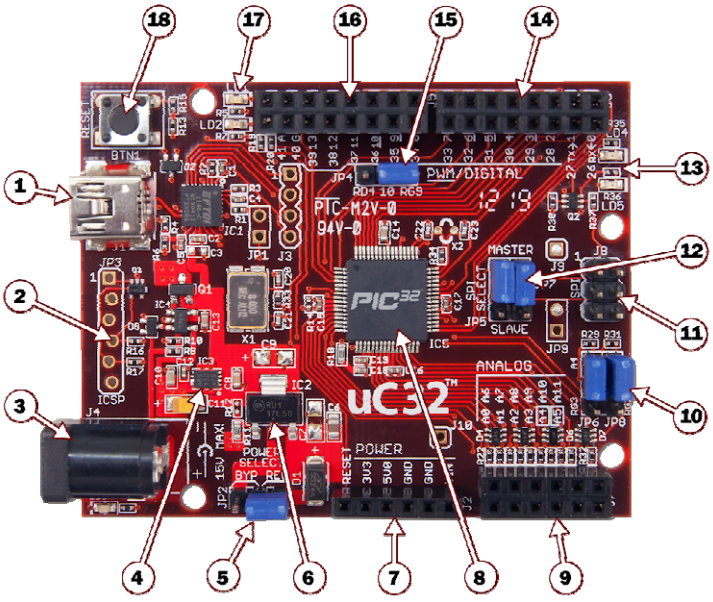


Obrázek 1.1: Všesměrový robot

2. chipKIT uC32

V této práci je pro vývoj dálkového ovládání použit vývojový kit uC32 (2.1) od firmy Digilent Inc. Tento kit je vyvinut tak, aby byl kompatibilní s Arduino kity. Je osazen mikroprocesorem PIC32MX340F512H (2.1), pro který tento kit nabízí více způsobů napájení. Lze jej napájet přes externí napájecí zdroj nebo pomocí USB. Vstupní napájení je regulováno dvěma regulátory napětí. První reguluje napětí na 5V a druhý na 3.3V. Kit je napájen 3.3V, ovšem poskytuje i 5V pro externí zařízení. Externím zařízením je myšleno např. displej, reproduktor, videokamera, atd. Pro LCD displej v této práci bylo použito napětí 3.3V. Z procesoru je na kit vyvedeno 6 portů, které umožňují přístup na konektory. Vývojový kit uC32 poskytuje celkem 42 IO pinů s podporou SPI (5), I2C, UART a PWM výstupem. Dále nabízí 12 IO analogových pinů, 2 LED a tlačítko reset. Kit je programovatelný přes prostředí MPIDE nebo pokročilé vývojové prostředí MPLAB ® IDE a programátor PICKit3.

V tomto projektu bylo použito prostředí MPLAB ® IDE. Důvodem pro zvolení tohoto prostředí byla plná kompatibilita s mikroprocesorem a také funkce obsažené v programu, které umožňují sledování a ladění programu.



Obrázek 2.1: Vývojový kit uC32

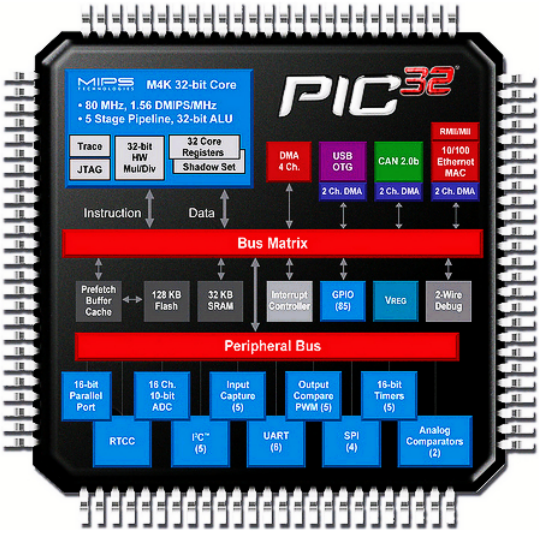
1 - USB konektor	7 - napájecí konektor	13 - 2 uživateleské LED
2 - konektor pro externí programátor	8 - mikroprocesor PIC32	14 - konektor pro přístup k digitálním pinům
3 - konektor pro externí napájení	9 - konektor pro přístup k analogovým pinům	15 - přepínač signálu (SPI/PWM)
4 - regulátor napájení na 3.3V	10 - přepínač signálu (I2C/analogový vstup)	16 - konektor pro přístup k digitálním pinům
5 - přepínač napájení	11 - SPI konektor	17 - stavové LED při komunikaci
6 - regulátor napájení na 5V	12 - SPI master/slave přepínač	18 - tlačítko RESET

Tabulka 2.1: Popis vývojového kitu uC32

2.1 Mikroprocesor PIC32MX340F512H

Jedná se o vysoce výkonné 32bitové mikrojádro (2.2)) firmy Microchip využívající Harvardskou architekturu. To znamená, že mikroprocesor má fyzicky oddělenou paměť pro data a paměť programu. Maximální taktovací frekvence mikroprocesoru je 80Mhz a počet operací až 1,65 DMIPS. Takto výkonné jádro poskytuje dostatek výkonu a paměti i pro náročné aplikace. Mikroprocesory PIC32 nám díky rychlému jádru umožňují rychlé přepínání kontextu a také odchycení přerušení. Jádro má 16 rychlých a přesných 10-bit kanálů ADC. Dále také obsahuje volitelné režimy RUN, SLEEP a IDLE, díky kterým lze řídit spotřebu. Mezi další vlastnosti, které nám jádro nabízí, patří např. rozhraní JTAG, WTD, 2 oscilátory (80MHz a 31kHz), 2 SPI rozhraní a 2 I2C sběrnice. Z jádra je celkem vyvedeno 64 pinů, které jsou připojeny na porty (2.1.1).

Mikroprocesor obsahuje 2 typy pamětí. První paměť je paměť typu Flash o velikosti až 512K, která slouží pro uchovávání samotného programu mikroprocesoru. Další paměť, kterou má mikroprocesor je paměť RAM s velikostí 32K. Ta slouží k uchovávání dočasných dat z výpočtů programu, ale také přes tuto paměť nastavujeme speciální registry. Těmi se nastavují integrované obvody čipu. Ke speciálním registrům se přistupuje přes jejich jména, pro zjednodušení jejich přístupu a přehlednosti kódu.



Obrázek 2.2: Blokové schéma PIC32

2.1.1 Porty mikroprocesoru

Použitý mikroprocesor obsahuje 6 portů. Neobsahuje pouze port A, ale dále již má zastoupeny všechny, tzn. B, C, D, E, F a G. Pomocí těchto portů lze přistupovat k jednotlivým pinům mikroprocesoru. Aby bylo možné přistupovat k pinům, je k portu přidružen registr TRIS. Ten určuje, zda se piny budou chovat jako vstupní nebo výstupní. Nastavení konfigurace pinu na vstup a výstup je provedeno tak, že pokud příslušný bit registru TRIS je nastaven na logickou nulu, pin je nastaven jako výstupní. V opačném případě pokud bude bit nastaven na logickou jedničku, bude pin vstupní.

Různé vlastnosti jednotlivých portů umožňují připojení různých periferních zařízení. Porty nemusí mít stejnou bitovou šířku, např. port B má šířku 16 bitů a port E šířku 8 bitů. Dalším rozdílem mezi porty je vlastnost připojení analogového vstupu, které umožňuje port B, ostatní porty umožňují pouze digitální vstupy.

Z tohoto důvodu byl port B použit pro připojení joysticků dálkového ovládání. K dalším pinům, které umožňují pouze digitální vstupy nebo výstupy byl připojen LCD displej a RF vysílač.

2.2 chipKIT Pmod Shield-Uno

Tento modul rozšiřuje chipKit uC32, kterému poskytuje další konektory pro použití periferních modulů. Pmod Shield-Uno je vstupně/výstupní modul. Je osazen pěti 2x6 Pmod konektory, které umožňují přístup na konektory modulu uC32. Dále se zde nachází, jeden SPI konektor a jeden I2C. Na modulu jsou i 4 uživatelské LED. Tento modul nelze použít samostatně, protože neobsahuje žádný mikroprocesor a slouží pouze jako rozšiřující modul. V této práci byl modul použit pro potřebu připojení RF modulu, ke kterému se přistupuje, pomocí SPI Pmod konektoru.

2.3 PmodRF2

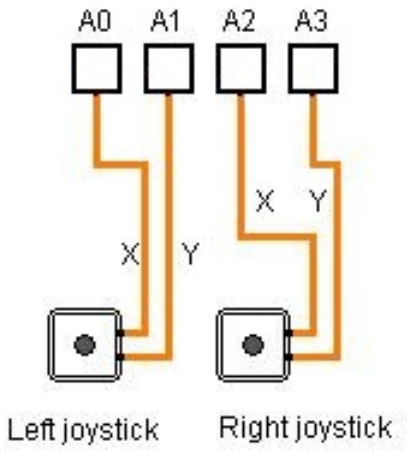
PmodRF2 je modul, který umožňuje jakékoliv řídící jednotce firmy Digilent rozšíření v podobě RF přijímače/vysílače. Na modulu je integrován RF vysílač MRF24J40 (2.3.1). Jedná se o vstupně/výstupní zařízení, tzn., že lze tento modul použít pro příjem i odesílání dat. Tato práce jej používá pro odeslání dat k řídící jednotce na průzkumném robotovi. Řídící jednotku průzkumného robota je potřeba správně nakonfigurovat a připojit k ní RF přijímač tak, aby byl robot schopný přijímat příkazy. To ovšem není součástí této práce.

2.3.1 MRF24J40

Jde o vysokofrekvenční vysílač s integrovanou anténou. Odpovídá standardu IEEE 802.15.4. Podpouje ZigBee, MiWi a MiWi P2P. Tento vysílač nabízí možnost volby vysílacího kanálu v rozsahu 2,405GHz - 2,480GHz. Obsahuje 4 vodičové SPI rozhraní, pomocí kterého probíhá konfigurace a dále odesílání a přijímání dat. Jeho součástí je HW mechanismus pro automatické potvrzení odezvy a CSMA-CA mechanismus. Tento mechanismus v praxi znamená, že vysílací zařízení před odesláním dat nejprve naslouchá, zda je přenosové médium volné. Pokud ano, jsou data odeslána. V opačném případě vysílací zařízení vyčkává na konec aktuálního přenosu a až poté odešle data. MRF24J40 obsahuje také HW 128bitové AES zabezpečení. AES je symetrická bloková šifra, což znamená, že pro šifrování i dešifrování je použit stejný klíč. Pro kontrolu detekce chyb při přenosu dat, je zde také v základním nastavení použita funkce CRC. CRC je nezávisle spočítán při odesílání dat a připojen k paketu. Následně je znovu vypočítán v zařízení, které data přijalo. Pokud kontrolní součty nesouhlasí, je zřejmé, že data byla při přenosu poškozena. Pro svou funkčnost nepotřebuje velké napětí a navíc umožňuje funkci SLEEP pro šetření energií. Je to vhodné řešení pro bezdrátové řízení, ať už se jedná o senzory, automatizaci či jako v této práci ovládání robota.

3. Joystick

Joystick je vstupní zařízení používané pro komunikaci s uživatelem. Díky snadnému a přesnému ovládání tato zařízení nalezla své uplatnění nejen v PC technice, ale i u průmyslových strojů, mobilních telefonů, atd. Joysticky se rozdělují na neproporciální (digitální) a proporciální (analogové). V této bakalářské práci byly použity dva proporciální joysticky z důvodu lepšího záznamu odchylky. Ty byly získány z analogového ovladače. Použitý joystick využívá dvou potenciometrů, díky kterým zaznamenává horizontální a vertikální pohyb, tedy pohyb po ose x a y.



Obrázek 3.1: Schéma připojení joysticků k pinům

Po připojení joysticku k napájení řídícího modulu s odporem 1K a připojení k uzemnění, bylo možné po softwarovém nastavení číst data z potenciometrů. Jedná se o analogové joysticky, a proto musely být připojeny k analogovým pinům (3.1). Jeden joystick je použit pro řízení směru a rychlosti průzkumného robota. Rychlost a směr robota je dán hodnotami na potenciometrech joysticku, to znamená, že čím více je joystick na ose vychýlen, tím větší rychlost robot získá daným směrem. Druhý joystick je zde použit pro rotaci robota na daném místě. Rychlost rotace není počítána, pouze podle vychýlení na ose x je určen směr otáčení robota. Rychlost otáčení je konstantní.

4. LCD displej

LCD jsou plochá zobrazovací zařízení s daným počtem pixelů. Tyto displeje nejsou náročné na energii, a proto je velmi vhodné jejich použití v bateriových přístrojích. Velkou výhodou těchto displejů je, že nemají škodlivý vliv na lidský zrak. U barevných LCD je každý pixel rozdělen do 3 subpixelů, to znamená, že je zde použit model RGB. LCD displeje se rozdělují na pasivní (STN) a aktivní (TFT). TFT znamená, že displej je tvořen drobnými tranzistory pro každý obrazový bod. Díky tomu je dosaženo velmi kvalitního obrazu a zamezuje se stínům při pohybu displeje.

V dálkovém ovládání v této práci byl použit 2,8"TFT LCD displej firmy HAOYU Electronics. Jedná se o displej s 65 tisíci barev s rozlišením 320*240 pixelů. Připojení displeje je možné buď pomocí SPI portu nebo pomocí 16bitového paralelního připojení. Zde bylo použito 16bitové připojení displeje, protože pomocí paralelního zapojení je dosaženo vyšší rychlosti přenosu dat. Pro možnost použití displeje je třeba dodat mu vstupní napětí. Displej obsahuje regulátor napětí, může být tedy napájen napětím 3,3V nebo 5V. IO porty pracují pouze s napětím 3,3V. Dále musí být připojeny piny pro příkazy zápisu, čtení, volbu čipu a volba instrukcí nebo dat. Jako poslední pin, který musí být připojen, je pin RESET, který zajišťuje, že displej, v případě potřeby ukončí svou činnost a začne znovu.

Displej využívá LCD ovladač SPFD5408 (4.1) a dotykový ovladač XPT2046. Je tedy možné jej použít i pro dotykové příkazy. V zadání práce bylo požadováno použití joysticků, a proto piny pro dotyk nebyly připojeny a nebyly naprogramovány příkazy pro čtení dotyku. V případě připojení pinů pro čtení dotyku na displeji a doprogramování příkazů pro jejich čtení by ovladač mohl fungovat i jako dotykové ovládací zařízení bez joysticků.

4.1 SPFD5408

SPFD5408 je 262144 barevný SoC ovladač pro malé a střední TFT LCD displeje. Podporuje až 240*320*RGB rozlišení, kterého lze dosáhnout volbou RAM pro grafická data. Tento ovladač má vestavěný časovač, který podporuje různé požadavky malých a středních přenosných displejů. SPFD5408 lze zapojit pomocí 8/9/16/18bitového paralelního rozhraní nebo přes SPI. Jak již bylo uvedeno, bylo použito 16-ti bitové paralelní připojení. Pomocí těchto připojení lze displej konfigurovat, nebo přistupovat k RAM paměti. Ovladač podporuje funkci zobrazování 8 barvami a pohotovostní režim pro řízení spotřeby.

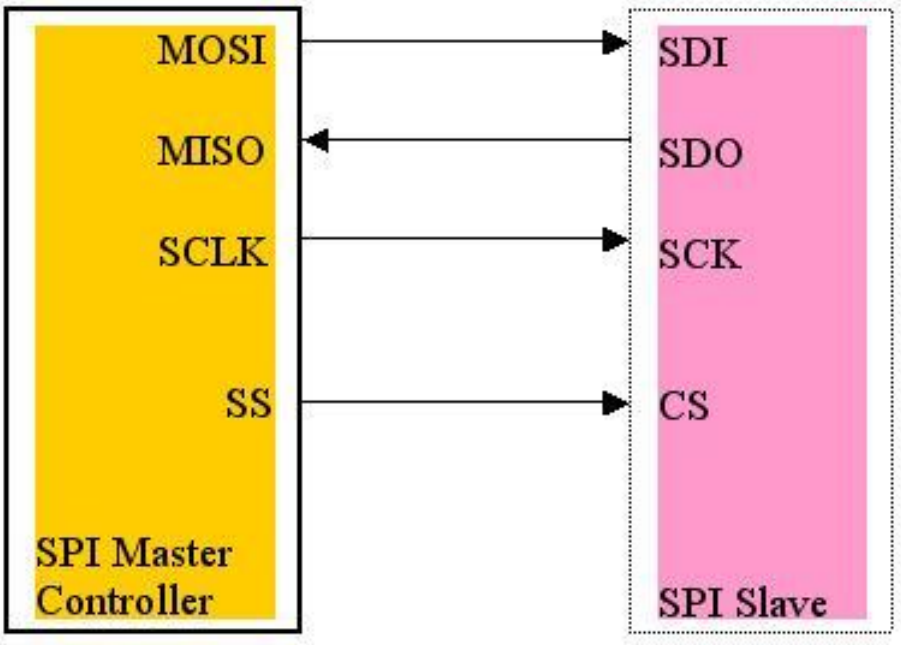
5. SPI

SPI protokol byl vyvinut firmou Motorola pro použití v mikroprocesorech. Tento protokol se používá při komunikaci mezi mikroprocesorem a ostatními obvody. Aby komunikace mohla probíhat, je třeba, aby jedno zařízení bylo nastaveno jako Master. To znamená, že toto zařízení zahajuje komunikaci a vysílá hodinové impulsy pro synchronizaci obvodů, které jsou Slave. Dále si volí zařízení Slave, se kterým bude komunikovat. Tuto volbu provádí přes SS, čímž zvolí Slave zařízení, které bude vysílat.

SPI sběrnice je složena ze 4 full duplex vodičů (5.1). Full duplex spojení znamená, že komunikace může probíhat v obou směrech současně. Vodiče sběrnice SPI jsou SCLK, MOSI, MISO a SS. Vodič SCLK je sériový hodinový signál. Hodinový signál je generován zařízením Master, kterým ovládá ostatní zařízení během komunikace. MOSI znamená, že data jsou přenášena ze zařízení Master, do zařízení Slave. Se stejným principem pracuje i MISO, jen v opačném směru. Posledním vodičem je vodič SS. Pomocí něj si zařízení Master zvolí, se kterým zařízením chce zahájit komunikaci. Volbu Slave zařízení přes pin SS je nutné využívat, pokud je připojeno více než jedno Slave zařízení. Pokud by zařízení Master nezvolilo, se kterým zařízením chce zahájit komunikaci, mohlo by dojít k odeslání dat na jiné zařízení. Volbu Slave zařízení není třeba provádět pokud je připojeno jen jedno Slave zařízení.

Pro zapojení Master-Slave zařízení existují dva způsoby. První možnost zapojení je zřetězení Slave zařízení. To znamená, že všechna Slave zařízení jsou připojena na na jeden SCLK pin a na jeden SS pin. Druhým typem zapojení je paralelní zapojení. Zde jsou také všechna Slave zařízení připojena na jeden SCLK pin, ale každému Slave zařízení je přiřazen jeden SS pin.

Mezi výhody SPI rozhraní patří především full duplex zapojení, vyšší propustnost než I2C protokol, libovolná velikost zprávy, jednoduché rozhraní, nízké požadavky na napájení a také to, že Slave zařízení nepotřebují vlastní oscilátory, jelikož jsou řízeny Master zařízením. SPI rozhraní má i nevýhody a velkou nevýhodou je, že Slave zařízení nijak nepotvrzuje spojení s Master zařízením.



Obrázek 5.1: Schéma zapojení SPI

6. Sestavení dálkového ovládání

Jak bylo již uvedeno dříve, není cílem práce pouze sestavit SW pro dálkové ovládání, ale celý jej zkonstruovat. Popis použitých modulů se nachází výše v práci. Dálkové ovládání tedy po sestavení obsahuje LCD displej pro zobrazování dat, joysticky pro řízení robota a RF modul. Tyto komponenty jsou připojeny k řídící jednotce a naprogramovány.

Jako první komponenta ovladače byly připojeny joysticky k mikroprocesoru. Ty jsou napájeny napětím poskytovaným z vývojového kitu. Je použito napětí 3.3V. Jelikož každý joysticky obsahuje dva potenciometry, bylo potřeba připojit jej každý k analogovému vstupu zvlášť. Nastavení pinů pro čtení analogových dat je popsáno v další kapitole.

Další komponentou je LCD displej. Tento prvek byl objednán z Číny od firmy HAOYU. SPI rozhraní pro připojení zde není využito, protože nelze pomocí něj získat takovou rychlost jako v případě paralelního zapojení. V tom případě je tedy nutné připojit 16 datových konektorů k mikroprocesoru. Displej je stejně jako joysticky napájen z vývojového kitu, ovšem lze jej napájet napětím 5V nebo 3.3V. Připojeny byly oba konektory, ale v této práci je použito napětí 3.3V. Napětí 5V nebylo využito, protože IO porty displeje využívají pouze 3.3V a pro tuto práci bylo dostačující napětí 3.3V. Dále jsou připojeny piny CSA, RS, RD a WR, aby bylo možné displej ovládat. Pomocí pinu RS se volí, zda se jedná o instrukce nebo o data, která jsou zaslána přes datové konektory. Piny RD a WR slouží pro nastavení čtení nebo zápisu dat na displej.

Posledním prvkem, který je potřeba pro sestavení dálkové ovládání je RF vysílač. Použit byl PmodRF2. Jelikož se jedná o stejného výrobce mikroprocesoru a tohoto modulu, stačilo pro připojení použít rozšiřující kit chipKIT Pmod Shield-Uno. Na tomto kitu se nachází konektor, přes který lze připojit RF vysílač.

7. Software dálkového ovládání

Mikroprocesor PIC32MX340F512H umožňuje naprogramování v jazyce C nebo ASM. Zde bylo využito jazyka C a vývojového prostředí MPLAB IDE. Toto vývojové prostředí musí být ovšem použito s programátorem pro nahrání zdrojového kódu do mikroprocesoru. Výhodou tohoto prostředí je možnost krokového ladění zdrojového kódu. Dále umožňuje automatické vyhledání manuálů k zvolenému přípravku, nastavení konfiguračních bitů a jejich vygenerování do zdrojového kódu a další užitečné věci pro pomoc při psaní kódu.

7.1 Konfigurace

První řádky zdrojového kódu jsou věnovány konfiguraci mikroprocesoru. Je zde nastaven primární oscilátor, který má frekvenci 8MHz. Výsledná frekvence po nastavení registrů je 80MHz. Nastavení oscilátoru je provedeno třemi příkazy. První příkaz FPLLIDIV je nastaven na hodnotu DIV_2. Tím je určeno, že vstupní frekvence bude vydělena 2. Její hodnota po tomto kroku je tedy 4MHz. Následující příkaz FPLLMUL je v této práci nastaven na hodnotu MUL_20, která vynásobí frekvenci 20x. Tím je nastavena na 80MHz. V posledním příkazu je nastaveno dělení frekvence 1x, což znamená, že hodnota oscilátoru se již nezmění. Sekundární oscilátor je vypnut. Ten má frekvenci 32,768kHz. Dalším nastavením je určeno, že se bude jednat o vysokorychlostní mód oscilátoru a také to, jaká bude dělička frekvence. V tomto případě je nastavena na hodnotu 8. Tím je ovlivněno taktování sběrnice periferií. Dále je zapnut Watchdog Timer, který hlídá, zda nedošlo k zaseknutí systému. Pokud ano, resetuje systém. Hlídání probíhá tak, že systém hlásí svůj průchod a pokud Watchdog není systémem v určitém časovém úseku vynulován, resetuje celý systém.

7.2 Main

Pro přehlednost zdrojového kódu je celý program tvořen v jednotlivých zdrojových souborech pro každý prvek ovladače. Všechny použité zdrojové kódy byly vytvořeny v průběhu této bakalářské práce. Pouze jeden zdrojový kód byl převzat z internetového úložiště, jedná se o knihovnu s ASCII znaky. Hlavní zdrojový soubor tedy obsahuje jen volání jednotlivých funkcí zdrojových souborů. Obsahuje tedy jen pár řádků kódu, na kterých zavolá hlavičkový soubor s konfigurací systému, načte kódy z ostatních zdrojových souborů a následně zavolá funkce pro inicializaci joysticků, displeje, SPI rozhraní a RF komunikaci. Předposlední funkce, které je vyvolána, je smazání displeje. Ta je zde pro případ, že by na displeji po zapnutí dálkového ovládání byla data, která zde zůstala po inicializaci. Dále již následuje funkce, která obstarává komunikaci mezi uživatelem a řídící jednotkou a dále k robotu.

7.3 Joystick

Ve zdrojovém souboru Joystick.c se nachází funkce pro inicializaci přerušení, vypnutí trasování a vypnutí ladícího rozhraní. Další funkce, která se zde nachází, je funkce pro nastavení analogového čtení dat. Toto nastavení je potřeba zavolat před prvním čtením dat. Pro nastavení čtení analogových pinů, je potřeba nastavit správně registry pro ovládání analogového portu (viz následující kód).

```
void initADC( int amask ){
        AD1PCFG = amask;
        AD1CON1 = 0x00E0;
        AD1CSSL = 0;
        AD1CON2 = 0;
        AD1CON3 = 0x1F3F;
        AD1CON1SET = 0x8000;
}
```

Příkazem AD1PCFG je určeno, zda jsou analogové vstupy nastaveny jako digitální. Pokud bude tedy proměnná amask nastavena jako 0, budou piny nastaveny jako analogové. V případě, že bude amask roven 1, budou piny nastaveny jako digitální. V registru AD1CON1 je touto hodnotou nastaveno, že data budou překonvertována po skončení čtení dat. Registr AD1CSSL je nastaven na hodnotu 0. To znamená, že nejsou zvoleny žádné kanály pro sekvenční skenování. V další konfiguraci je nastaven registr AD1CON2 také na hodnotu nula. Tím je určeno použití multiplexoru A. Dále se zde nachází nastavení maximálního času vzorkování a jako poslední je nastaven registr AD1CON1SET. Tímto posledním krokem je zapnut ADC.

Funkce readADC pak již čte data na konkrétním pinu. Čtení probíhá tak, že je nastaven analogový kanál, který se má přečíst, zapne se vzorkování dokud nebude hotova konverze, poté je vrácena hodnota, která se nachází ve vyrovnávací paměti. Je

zde také ještě vytvořena funkce pro pozdržení mikroprocesoru o X milisekund. V této funkci je zapnut časovač, který při každém přetečení navýší hodnotu proměnné wait. Dokud bude proměnná wait menší než námi zadaná hodnota, bude se vykonávat tento proces. Tím je zaručeno pozdržení mikroprocesoru o uživatelem zadaný počet milisekund.

V hlavičce tohoto zdrojového kódu jsou definovány konstanty funkce pro čtení jednotlivých oscilátorů joysticků. Jsou pojmenovány podle pozice na ovladači a osy, kterou načítají. Jsou to tedy konstanty LeftY, LeftX, RightY a RightX. Dále je zde definice nastavení pro zapnutí a vypnutí časovače a hodnota časovače. Poslední co lze v hlavičkovém souboru nalézt je definice funkcí zdrojového kódu a načtení potřebných hlavičkových souborů.

7.4 LCD

Dalším souborem v práci je soubor LCD.c. Jak již z názvu vyplývá, zdrojový kód obsažený v tomto souboru se zabývá nastavením a základními příkazy pro LCD displej. Při základním nastavení displeje jsou nejdříve nastaveny piny vývojového kitu připojené k displeji. Všechny tyto piny musí být nastaveny jako digitální a výstupní. To lze jednoduše provést příkazem mPORTXSetPinsDigitalOut(BIT_Y), kde X zastupuje označení portu a Y číslo pinu.

Po nastavení pinů následuje výčet příkazů pro nastavení displeje, aby jej bylo možné používat. Toto nastavení probíhá tak, že na adresu v registru LCD displeje je zapsána hodnota, podle které displej přizpůsobí své chování. Nastavení zahrnuje například nastavení orientace dat v displeji, změnu velikosti obrazu, nastavení prázdných řádků v záhlaví a zápatí, nastavení kmitočtu, barev, šířky a výšky zobrazovacího okna a mnoho dalších potřebných nastavení. Toto nastavení bylo zvoleno dle katalogového listu k produktu [8].

Správné přiřazení dat na piny je další nezbytná část pro korektní zapsání adresy či dat na displej. V souboru LCD.c se tedy nachází dvě funkce - Set_Low_bit a Set_High_Bit. V těchto funkcích je určeno jak budou data přiřazena na jednotlivé piny konektoru. Tyto funkce jsou používány jak při konfiguraci displeje, tak při zasílání dat na displej. O správné rozdělení dat, která jsou zaslána na dolních 8 bitů displeje a která na horních 8 displeje se stará funkce LCD_Send. Tato funkce pomocí maskování bitů a jejich posunu rozdělí vstupní data na horní a dolní bity a následně zavolá příslušnou funkci pro přiřazení dat na piny.

Pro možnost rozlišení, zda se zapisují data nebo adresa, jsou vytvořeny funkce LCD_WriteIndex (viz následující kód) a LCD_WriteData.

```
static void LCD_WriteIndex(uint16_t index) {
        LCD_CS(0); /* CS low */
        LCD_RS(0); /* RS low */
        LCD_RD(1); /* RD high */
        LCD_WR(0); /* WR low */
        /* write data */
```

```
        LCD_Send(index);

        LCD_WR(0); /* WR low */
        LCD_WR(1); /* WR high */
        LCD_CS(1); /* CS high */
}
```

Jak lze z kódu vidět, nejdříve probíhá volba čipu, která následuje zvolením zápisu do registru. V případě, že by zde byl příkaz LCD_RS(1), znamenalo by to, že bude zapisována na předem zapsanou adresu hodnota v proměnné index. V tomto případě je zde zdrojový kód pro zápis adresy, proto je v příkazu LCD_CS hodnota 0. Dále musí být vypnuto čtení dat a povolen zápis. Po tomto nastavení již může být zapsána hodnota v proměnné index. Ta v tomto případě zastupuje adresu v registru displeje. Algoritmus funkce LCD_Send, byl již popsán výše. Po zapsání hodnoty je provedena ještě ukončovací rutina zápisu dat.

V tomto souboru pro displej jsou vytvořeny ještě další funkce, pomocí kterých lze displej vymazat nebo kreslit základní tvary jako jsou čára, kruh, trojúhelník nebo šipka. Jsou zde i dvě funkce pro text. Jedna funkce je pouze pro jeden znak z ASCII tabulky a nazývá se PutChar. Druhá funkce je určena pro textové řetězce a nazývá se GUI_Text.

Aby bylo možné používat tyto funkce pro znaky, je v programu použit soubor s názvem ASCII.c, který byl nalezen na internetovém úložišti se zdrojovými kódy. Ve zdrojovém kódu byl samozřejmě zachován autorův podpis. V SW řešení je tento zdrojový kód zachován beze změn. SW dálkového ovládání jej využívá při jakékoliv práci s textem. V literatuře v této práci je uveden internetový odkaz na zdrojový kód s ASCII tabulkou a jejím převodem [13].

7.5 SPI

Dalším úkolem, který je nezbytný pro tuto bakalářskou práci, je vytvoření SPI komunikace. Ta je později využita při bezdrátové komunikaci. Výše v této práci byly popsány výhody a nevýhody tohoto rozhraní a také způsob jak spolu dvě zařízení přes něj komunikují.

Mezi zdrojovými soubory programu se tedy nachází i soubor SPI.c., ve kterém je zdrojový kód pro nastavení tohoto rozhraní v zařízení. Jsou zde vytvořeny funkce pro nastavení Master i Slave zařízení. Dále se zde ještě nachází funkce pro čtení a zápis do vyrovnávací paměti rozhraní SPI.

Následující zdrojový kód obsahuje nastavení pro Master zařízení. Pro nastavení Slave zařízení je zdrojový kód velmi podobný, změny v kódu budou popsány.

```
void SPI_Initialization_Master() {
        char dummy;
        CS = 0;
        SCK = 0;
        MISO = 1;
```

```
        MOSI = 0;
        SPI2CONbits.ON = 0;
        dummy = SPI2BUF;
        SPI2BRG = 1999999;
        SPI2CON = 0x8120;
}
```

Na prvním řádku je vytvořena proměnná dummy, která bude použita později. Následující řádky obsahují nastavení pro Master zařízení. Proměnné CS, SCK, MISO a MOSI jsou nastaveny v hlavičkovém souboru a zastupují registr TRIS pro příslušné piny. Pin CS, který slouží pro volbu Slave zařízení, je jako výstupní. Stejně tak i pin SCK zastupující pin pro synchronizaci komunikujících zařízení. Dále je nastaven pin MISO jako vstupní, protože přes tento pin zařízení master přijímá data od zařízení Slave a jako poslední je nastaven pin MOSI jako výstupní. Přes tento pin jsou zasílána data do zařízení Slave.

Pokud by tento zdrojový kód měl být použit v zařízení Slave, je potřeba hodnoty pro piny CS, SCK, MISO a MOSI nastavit opačně. Dalším příkazem je vypnuto rozhraní SPI, aby byl vynulován jakýkoliv předchozí stav tohoto rozhraní. Nyní je použita proměnná dummy, kterou je přečtena vyrovnávací paměť tohoto rozhraní a tím je zajištěno, že neobsahuje žádná data.

Registrem SPI2BRG je nastaveno, jaká bude frekvence SPI. Výsledná hodnota je vypočítána dle jednoduchého vzorce (7.1).

$$Fsck = \frac{Fpb}{2 * SPIxBRG + 1} \tag{7.1}$$

Neznámá hodnota X v této rovnici pouze nahrazuje číslo použitého registru pro příslušné SPI rozhraní. V této práci zde bude hodnota 2, protože je použito rozhraní SPI 2. Hodnota, která je zde přiřazena do registru SPI2BRG je 1 999 999. Po dosazení do uvedeného vzorce lze vypočítat, že SPI frekvence bude odpovídat 20 MHz, to je maximální rychlosti přenosu tímto rozhraním.

Do registru je SPI2CON přiřazena hexadecimální hodnota 0x8120. První hodnota 0 je zde proto, že prvních 5 bitů registru SPI2CON neobsahuje žádné konfigurační bity. Hodnota 2 odpovídá binárnímu zápisu 0020b. První bit je nepotřebný, protože se jedná o pátý bit registru SPI2CON. Druhému bitu je přiřazena 1, která zapíná Master mód zařízení. V případě použití v zařízení Slave by zde byla také hodnota 0. Na dalším bitu je opět 0. Na této pozici konfiguračních bitů registru je bit CKP. Hodnota 0 na tomto bitu znamená, že pokud bude hodinový signál v klidovém režimu, bude nastaven na nízkou úroveň. Další konfigurační bit, který je zde přiřazen na hodnotu 0 je bit SSEN. Tím je nastaveno, že volba CS pinu bude ovládána portem zařízení. Třetí hexadecimální hodnota je 1, kterou lze v binárním zápisu zapsat jako 0001b. První bit je zde tedy nastaven na 1. Na této pozici se nachází konfigurační bit CKE. Ten ovlivňuje, zda budou data čtena při náběžné nebo sestupné hraně. Jelikož je bit CKP nastaven na hodnotu 0 a bit CKE na hodnotu 1, data budou čtena při sestupné hraně. Další konfiguračním bitem je bit SMP. Tomu je zde přiřazena hodnota 0. Ta určuje, kdy budou data odeslána. V tomto případě je nastavena hodnota 0, to znamená, že se začnou odesílat data uprostřed přečtených

dat. Další dva konfigurační bity jsou určeny pro změnu módu přenášených dat. Oba dva bity jsou nastaveny na hodnotu 0, a proto je velikost přenášených dat 8 bitů. Poslední hexadecimální hodnota je 8, v binárním zápisu 1000b. Tím je zapnuto SPI.

V souboru SPI.c je ještě funkce pro zápis a čtení dat. Tato funkce je univerzální, lze ji tedy použít pro čtení i pro zápis. Je pojmenována RW_SPI a přejímá parametr typu integer. Proměnná data je nejdříve zapsána do vyrovnávací paměti SPI2BUF. Následuje cyklus while, který trvá tak dlouho, dokud není bit SPIRBF roven hodnotě 0. Hodnota 0 znamená, že přenos dat byl dokončen. Následuje přečtení a navrácení hodnoty z vyrovnávací paměti, kterou odeslalo Slave zařízení. Protože nejsou očekávána žádná data od Slave zařízení, je tato hodnota bezvýznamná.

7.6 RF

Po zprovoznění SPI rozhraní je možná konfigurace a přenos dat pomocí RF modulu. Ve zdrojovém souboru pro tento modul jsou vytvořeny metody pro jeho nastavení, čtení a zápis dat, zápis dat na krátkou a dlouhou adresu registru a také pro čtení z krátké a dlouhé adresy registru. Pro nastavení tohoto modulu jsou vytvořeny v hlavičce definice adres dle katalogového listu [11]. Při konfiguraci modulu jsou zde použita nadefinovaná jména adres.

Jak již bylo zmíněno, pro nastavení tohoto modulu je použito SPI rozhraní. Jeho nastavení musí být jako Master, protože tento modul s řídící jednotkou komunikuje jako Slave zařízení. Z toho vyplývá, že jak v dálkovém ovládání, tak i v robotu musí být použito nastavení SPI jako Master zařízení.

První kroky nastavení jsou dle doporučení z katalogového listu věnována HW a SW resetu. Pro HW reset je na reset pin modulu zapsána 0. Poté se vyčká 100ms a na reset pin je zapsána 1 a opět se vyčká 100ms. Následně je zapsána na určenou adresu v registru logická 1 pro reset stavu a následně logická 0. Po tomto resetu dojde k nové kalibraci obvodu, ale data v registru nejsou změněna. Následuje SW reset, při kterém dojde k resetu veškerého předchozího nastavení a resetu MAC adresy.

Při konfiguraci je v následujících příkazech zapsána MAC adresa. V katalogovém listu [11] je uvedena doporučená konfigurace systému. Dle tohoto doporučení je provedeno nastavení ve zdrojovém kódu.

Mezi prvními příkazy nastavení je zapnuta FIFO fronta pro data, nastaven čas, který uběhne před každým odeslaným paketem, doba pro ustálení bitů a mezera mezi jednotlivými bity. Následuje zapnutí PLL, který je nezbytný jak pro příjem, tak pro odesílání dat. PLL je systém, který generuje výstupní signál. Ten má fázi závislou na vstupním signálu. Tento obvod porovnává tyto fáze a provádí taková změny, aby se fáze shodovaly.

Velmi důležitým nastavením RF modulu je určení toho, kdo bude řídit komunikaci a kdo bude jen ovládaným zařízením. Ve zdrojovém kódu je vytvořena funkce pro nastavení koordinátora komunikace i závislého zařízení. Pro toto dálkové ovládání je použita funkce pro nastavení koordinátora. To znamená, že zařízení, které se bude připojovat k tomuto ovladači, musí použít funkci pro nastavení závislého zařízení.

Poté následuje nastavení RSSI. To je počítáno pro každý paket. Dále dojde k zapnutí všech přerušení a nastavení kanálu 11. Kanál 11 je volba frekvence o velikosti 2405MHz. To je minimální možná hodnota frekvence tohoto modulu. V posledním kroku nastavení je příkaz pro použití plné energie. Následně je proveden reset s novým nastavením.

Pro zápis dat na adresu v registru slouží dvě funkce - WriteShort pro zápis na krátké adresy a WriteLong na dlouhé adresy registru. Rutina pro zápis na adresu je následující:

- jako první krok je na SPI bit pro CS nastavena 0
- je zadaná adresa posunuta o jeden bit vlevo
- použití logického součinu s hodnotou 0x7E
- aplikování logického součtu výsledné hodnoty a hodnoty 0x01

Výsledkem je cílová adresa v registru. Tato hodnota je pomocí SPI rozhraní zapsána do registru modulu. Dalším krokem je zápis zadaných dat. Předchozí krok zajistí, že data jsou zapsána na požadovanou adresu v registru. Poslední krok nastaví na CS bit 1. Tím je zapisovací rutina u konce.

Při zápisu na dlouhou adresu registru je zapisovací rutina velmi podobná. Rozdíl je v tom, že k získání výsledné adresy je použit jiný postup výpočtů. Vzhledem k tomu, že se jedná o dlouhou adresu, probíhá její zápis ve dvou krocích. Poté co je zapsána celá adresa, následuje stejný postup jako v předchozí funkci.

Stejně jako pro zápis, tak i pro čtení z krátké a dlouhé adresy registru, zdrojový kód obsahuje dvě funkce. Obě funkce jsou si také velmi podobné, jako je tomu u zápisu dat. U dlouhé adresy je opět ve dvou krocích nastavena adresa čtení.

Rutina pro čtení dat obsahuje opět v prvním kroku nastavení 0 na CS bit, dále přepočet a zápis adresy a pak již přečtení dat. Poté je opět na CS bit nastavena 1 a v posledním příkazu je získaná hodnota z adresy registru vrácena předchozímu volání funkce.

Poslední dvě funkce ve zdrojovém kódu jsou věnovány posílání a přijímání dat. Funkce pro přenos dat jsou nazvány Send pro zaslání a Receive pro jejich přijetí.

Zdrojový kód pro funkci Send vypadá následovně:

```
void Send(uint8_t *data, uint8_t length) {
        uint8_t i;
        WriteLong(0x000, 0);
        WriteLong(0x001, length);
        for (i = 0; i < length; i++)
                WriteLong(0x002 + i, data[i]);

        WriteShort(TXNMTRIG, 0x01);
}
```

Funkce Send přejímá jako první parametr data, což je ukazatel na adresu v paměti, kde jsou data k odeslání uložena. Druhým parametrem je délka přenášených dat. Prvním příkazem je vytvořena proměnná i, která je použita ve for cyklu pro iteraci a kontrolu počtu cyklů a také jako index v zasílaných datech. První příkaz slouží k zápisu hodnoty 0 na dlouhou adresu registru. Tím je nastaveno, že nebude v hlavičce žádná adresa. Druhý příkaz slouží k zapsání délky zasílaných dat. Následuje cyklus, ve kterém jsou od adresy 0x002 postupně zapisována data tak, jak jsou uložena v paměti za sebou. Na prvních 8 bytech zasílaných dat jsou data, která vypadají jako platná hlavička paketu. Uživatel je tedy nemusí nastavovat, jsou zde pevně uložena. Po těchto 8 bytech následují zasílaná data. Data v paměti mohou být maximálně 1 byte a maximální rozsah pole je 128 bytů. Pro zasílaná data je tedy vyhrazeno 120 bytů. Poslední příkaz v této funkci slouží k zahájení přenosu dat. Tento bit je HW automaticky navrácen na hodnotu 0.

Poslední funkcí v tomto zdrojovém souboru pro přenos dat je funkce Receive. Její zdrojový kód je zobrazen zde:

```
uint8_t Receive(uint8_t *data, uint8_t maxLength) {
        uint8_t i, length;
        uint8_t lqi, rssi;
        if (ReadShort(INTSTAT_M)& 0x08) {
                length = ReadLong(0x300);
                lqi = ReadLong(0x301 + length);
                rssi = ReadLong(0x302 + length);
                for (i = 0; i < length; i++){
                    if (i < maxLength)
                      *data++ = ReadLong(0x301 + (uint16_t) i);
                    else
                      ReadLong(0x301 + (uint16_t) i);
                }
                if (length < maxLength)
                    return length;
        }
        return 0;
}
```

Tato funkce pro přijetí dat vyžaduje dva parametry. První parametr slouží k získání adresy v paměti, na kterou se ukládají přijatá data a druhý k zadání maximálnímu počtu dat, která mohou být uložena na získanou adresu v paměti.

První dva řádky zdrojového kódu vytvářejí proměnné pro další použití. Následuje podmínka, pomocí které je přečtena adresa s bity pro přerušení. Zde se program zajímá o bit, který udává, zda jsou nějaká přijatá data v paměti. Pokud je podmínka vyhodnocena záporně, je navrácena hodnota 0. V opačném případě je získána délka přenesených bytů, hodnota kvality spojení a síla signálu. Následuje for cyklus. Zde je využito proměnné i. Tato proměnná opět slouží k iteraci v cyklu a také jako ukazatel na adresu, ze které se mají číst přijatá data. Tuto proměnnou ještě v každém cyklu

vyhodnocuje podmínka, která zjistí, zda nepřesáhla maximální možný počet uložených hodnot na adresu v paměti. Pokud je proměnná i větší než maximální množství dat, nejsou data po přečtení uložena. V opačném případě jsou data ukládána na adresu v paměti. Po skončení cyklu následuje poslední vyhodnocení a to, zda je délka přenesených dat menší než maximální počet dat, která bylo možné uložit. Pokud je vyhodnocení kladné, je vrácena hodnota odpovídající délce přenesených dat.

7.7 Hlavní program

Tato část SW využívá všechny dříve popsané metody a funkce pro jednotlivé prvky dálkového ovladače. Tento algoritmus tvoří s použitím zdrojových kódů ze zdrojových souborů SW dálkového ovladače. I zde jsou vytvořeny pomocné funkce pro přehlednost zdrojového kódu a pro zamezení duplicity kódu.

Na začátku tohoto algoritmu je základní nastavení proměnných. Následuje vykreslení všech prvků na displej. Těmito prvky se rozumí menší kruh v levém horním rohu a uvnitř kruhu z jeho středu směrová šipka s orientací vpřed. Tento prvek bude sloužit k zobrazení, jak je robot otočen od původního stavu. Dále je uprostřed displeje vykreslen kruh, ve kterém bude zobrazován směr a rychlost robota. Nad joysticky jsou vykresleny ukazatele, kam lze jednotlivé joysticky vychylovat, aby byla získána data.

Po vykreslení všech prvků na displej, je zavolána funkce initADC, která se nachází ve zdrojovém souboru JOYSTICK.c. Této funkci je předána hodnota 0, kterou je nastaveno, aby piny byly analogové. Poté již následuje cyklus, ve kterém se nachází rutina pro ovládací zařízení.

První řádky tohoto cyklu jsou věnovány vynulování proměnných, do kterých jsou ukládána data, která jsou poté odeslána robotu. Jedná se o proměnné action, direction, value_x a value_y. Proměnná action udržuje typ pohybu, buď je zde uložena hodnota 2 pro rotaci, hodnota 1 pro pohyb robota nebo 0 pokud není proveden žádný pohyb. Do proměnné direction se ukládají data ohledně směru pohybu, v případě rotace se jedná o hodnotu 8 při rotaci vlevo a 1 pro rotaci vpravo. Při pohybu robota ještě může být přičtena hodnota 4 při pohybu vpřed nebo hodnota 2 při pohybu vzad.

Dále je přečtena hodnota z levého joysticku. Pokud tato hodnota nebude odpovídat střední hodnotě potenciometru pro osu X na levém joysticku, program vyhodnotí, že joystick je vychýlen od svého středu. Na základě tohoto vyhodnocení se odvíjí jeho další činnost. Touto činností je myšleno, že program ještě dle hodnoty potenciometru rozpozná, na kterou stranu je joystick vychýlen, zda nalevo nebo napravo od svého středu. Dle tohoto rozhodnutí je zvolena vhodná část algoritmu, do které se program dostane po tomto rozhodnutí. Oba dva zdrojové kódy jsou podobné, ať je joystick vychýlen nalevo nebo napravo. V obou případech se jedná o rotaci robota. Změna je ovšem hned v prvních řádcích kódu, zde se počítá úhel otočení. V případě vychýlení joysticku vpravo je zde krok otočení od předchozí hodnoty 2°, v druhém případě je zde krok -2°. Následně je smazán ukazatel rychlosti a směru robota. Jelikož je robot otáčen, nemá rychlost ani směr. Poté jsou spočítány nové

hodnoty pro vykreslení ukazatele otočení a jeho překreslení do nových hodnot. Nové souřadnice bodů jsou vypočítány dle rovnic:

$$x = \cos(\frac{uhel * \pi}{180}) * (x - stred_x) - \sin(\frac{uhel * \pi}{180}) * (y - stred_y) + stred_x \quad (7.2)$$

$$y = \sin(\frac{uhel * \pi}{180}) * (x - stred_x) + \cos(\frac{uhel * \pi}{180}) * (y - stred_y) + stred_y \quad (7.3)$$

Proměnná uhel ve vzorci zastupuje úhel otočení od původního směru, tzn. od směrové šipky ukazující vpřed. Proměnná x a y jsou souřadnice bodu, který má být otočen. Zbývající proměnné stred_x a stred_y jsou souřadnice, okolo kterých je bod rotován.

Poté již následuje přiřazení hodnot do proměnných, které jsou odeslány k robotu. Do proměnné action je uložena hodnota 'ROT'. Ta určuje, že robot je rotován a že následující hodnotu určují pouze hodnoty pro rotaci. Do proměnné direction je dle směru rotace uložena hodnota 8 pro levou rotaci nebo hodnota 1 pro pravou rotaci. Proměnné value_x a value_y obsahují vždy procentuální vyjádření rychlosti v daném směru, protože se zde jedná o rotaci a jak již bylo řečeno, rychlost je při ní konstantní. To je důvod, proč je do proměnné value_x přiřazena hodnota 100. Do proměnné value_y je zde přiřazena hodnota 0, jelikož není využita tato hodnota při rotaci.

V případě, že program určí levý joystick jako nevychýlený, je programu umožněn vstup do jiné části algoritmu, než tomu bylo v předchozím případě. Rozhodování o vychýlení a směru vychýlení probíhá pomocí porovnání v podmínkách.

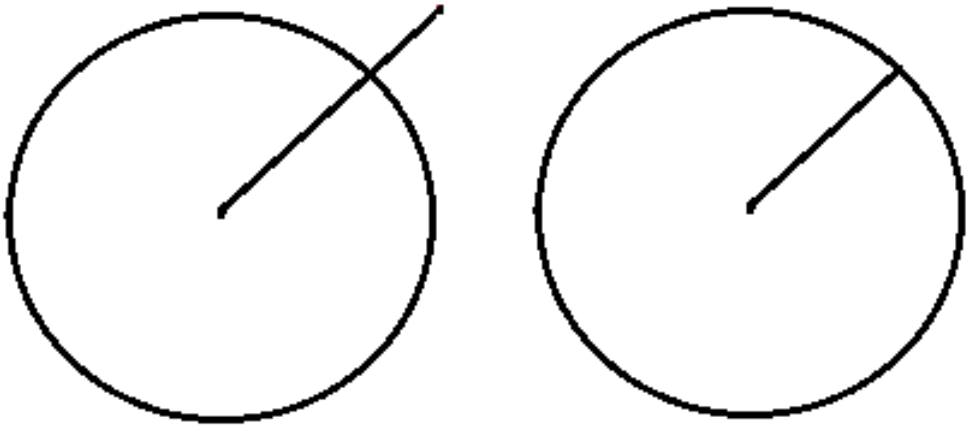
Tato část algoritmu pracuje především se 4 proměnnými hodnotami. Proměnné new_x a new_y zastupují nové souřadnice a proměnné old_x a old_y předchozí souřadnice. Těch je využito pro kontrolu, zda došlo ke změně souřadnic. Pokud ne, program nemusí znovu vykreslovat ukazatel. Pokud by se alespoň jedna souřadnice změnila, program dostane příkaz o překreslení ukazatele.

Na začátku tohoto algoritmu tedy proběhne kontrola těchto souřadnic a případné překreslení. Poté následuje přiřazení nových souřadnic do starých pro další kontrolu. Poté jsou načtena data z potenciometrů pravého joysticku z osy X a Y. Po načtení dat jsou data přepočítána do rozsahu 0 - 100. Tento rozsah procentuálně vyjadřuje vychýlení a určuje cílové souřadnice směrového ukazatele.

Přepočet probíhá ve funkci tak, že je nejdříve rozhodnuto, kterým směrem došlo k vychýlení. To je důležité, protože pokud hodnota z potenciometru je větší než jeho střední hodnota, je od načtené hodnoty odečtena středová, v druhém případě je tomu naopak. Po tomto odečtení je ještě hodnota vydělena hodnotou odpovídající 1/100 rozdílu maximální hodnoty potenciometru a střední hodnoty, případně střední hodnoty a minimální hodnoty potenciometru. Nakonec je tato hodnota převedena na celé číslo a navrácena do proměnných.

Poté co jsou tyto přepočty dokončeny, jsou známy souřadnice pro směrový ukazatel a procentuální vychýlení joysticku po osách. Následuje tedy výpočet rychlosti.

Pro výpočet rychlosti byla použita Pythagorova věta. Při použití trojúhelníku v tomto případě tvoří souřadnice odvěsny pravoúhlého trojúhelníku a rychlost odpovídá přeponě. Hodnota rychlosti je počítána, protože docházelo k překročení hranice rychlosti. Tato chyba vznikla, protože pokud byl joystick nesměrován např. v úhlu 45°, potenciometry os navracely 100% vychýlení. To po přepočtu na souřadnice znamenalo krajní bod na ose x a y a směrový ukazatel byl vykreslen do těchto souřadnic (7.1).



Obrázek 7.1: Vykreslení směrového ukazatele

Na levém obrázku je vidět, jak byl ukazatel vykreslen. Tento nechtěný jev byl tedy dále řešen tak, aby byl získán obraz, jaký je na obrázku vpravo. Pro získání správného obrazu bylo třeba přepočítat souřadnice. Přepočet proběhne pouze v případě, že se souřadnice nerovná nule. Pokud ne, je zavolána funkce Coord, které jsou předány parametry souřadnic osy x a y a vypočtená rychlost. Přestože jsou předány souřadnice x a y, dle pořadí těchto souřadnice je určeno, jaká souřadnice bude přepočítána. Po alokaci proměnných je uložena hodnota přepočítávané souřadnice. Tato hodnota později bude sloužit k určení, zda byla hodnota kladná nebo záporná. Poté jsou souřadnice vyděleny hodnotou 10. Následuje dělení rychlosti součtem předchozích hodnot. V předposledním kroku je hodnota tohoto dělení vynásobena předem vydělenou souřadnicí. Tím je získána přepočtená souřadnice. Ještě je ale třeba navrátit jí kladné nebo záporné znaménko. To lze rozhodnout dle dříve uložené hodnoty. Po tomto kroku je již přepočet souřadnice kompletní. Tento krok je proveden x a y souřadnicemi.

Následuje přiřazení hodnot pro robota. Hodnoty v proměnných value_x a value_y musely být přiřazeny před provedením přepočtu souřadnic. Důvodem toho je, že by po přepočtu v určitém směru nikdy nenabývaly 100% a robot by tedy nejel plnou rychlostí. Další přiřazení do proměnných ale probíhá až zde na konci algoritmu.

Pokud by byly souřadnice nulové, do proměnné action je přiřazena hodnota 0, v opačném případě hodnota 1. Poté je do proměnné direction přiřazena hodnota dle souřadnic. Pomocí podmínek systém vyhodnotí vychýlení joysticku a na základě vyhodnocené podmínky uloží hodnotu do proměnné.

Poté jsou data uložena do pole dat, které je odesláno robotu. Jak již bylo zmíněno, prvních 8 bytů je vyhrazeno pro hlavičku paketu, data se tedy ukládají až od 9. bytu. Na 9. byte je uloženo 6 bitové ID dálkového ovládání pro jeho identifikaci, na zbylé dva bity je uložen typ pohybu robota z proměnné action. Další byte obsahuje informace o směru pohybu. Na 11. a 12. bytu paketu se nachází velikost hodnot na ose y a x. Toto ukládání probíhá opakovaně v každém cyklu. Následuje přenos dat, který je popsán v kapitole RF (7.6). Poslední příkaz algoritmu odešle data pomocí RF přenosu robotu ke zpracování a celý cyklus se opakuje.

Závěr

Cílem této bakalářské práce bylo sestavit prototyp dálkového ovládání a vytvořit jeho SW. Tento dálkový ovladač měl obsahovat dva joysticky, LCD displeje a měla být navržena koncepce přenosu dat přes RF modul. Ovladač měl být navržen pro řízení všesměrového robota.

K tomu, aby tato práce mohla vzniknout, bylo třeba seznámit se s mikroprocesorem, jeho vlastnostmi a způsobem jeho programování. Dále bylo třeba nastudovat katalogové listy pro LCD displej a jeho čip a také pro RF modul. Bez těchto znalostí by nebylo možné tuto práci dokončit.

Psaní zdrojového kódu pro mikroprocesor PIC32MX340F512H probíhalo v prostředí MPLAB ® IDE. V tomto vývojovém prostředí je v jazyce C naprogramován celý SW pro dálkové ovládání. Programování mikroprocesoru probíhalo pomocí programátoru PICKit3. První řádky zdrojového kódu na začátku celé práce byly věnovány k seznámení s příkazy pro mikroprocesor. Po získání znalostí o základních příkazech a způsobu jak pracovat s mikroprocesorem, bylo dalším krokem HW sestavení dálkového ovládání.

Sestavení tohoto prototypu zahrnovalo připájení pinů potenciometrů a pinů LCD displeje k pinům mikroprocesoru. Poté následovalo sestavení SW pro práci s jednotlivými prvky ovladače. Nejtěžším úkolem této práce bylo správně pochopit katalogový list a poté naprogramovat SW pro LCD displej. Za pomoci katalogových listů a doporučené literatury byl nakonec sestaven SW pro tyto prvky tak, aby bylo možné jejich použití v práci.

Po sestavení SW pro prvky ovladače následoval vývoj algoritmu, který by realizoval dálkové ovládání. Postupně byl sestavován a laděn algoritmus, který v konečné verzi realizuje čtení dat z joysticků a vykreslování dat na displej dle daných parametrů.

Posledním krokem této bakalářské práce bylo vytvořit koncept přenosu dat přes RF modul. Tento modul pochází od stejného výrobce jako vývojový kit uC32 s mikroprocesorem. Nebylo tedy třeba nic pájet, stačilo pouze připojit modul do konektoru a vytvořit SW pro jeho nastavení a odesílání dat. Po přečtení katalogového listu [11] a nastavení daných hodnot dle doporučení, z něj byl vytvořen koncept pro přenos dat. Pomocí tohoto konceptu přenosu dat jsou přes tento modul zasílána data k dalšímu nastavenému zařízení. V tomto případě byl přenos vyzkoušen na stejném mikroprocesoru s přidaným chipKIT Basic I/O Shield modulem. Na něm byly podle pohybů joysticky na ovladači rozsvěcovány LED diody, tím bylo ověřeno, zda jsou data přenesena správně. To je ukazatel funkčnosti modulu a SW konceptu.

Dalším možným pokračováním v této práci by bylo rozšíření průzkumného robota

o čidla či nějaký pohybový prvek. Z čidel robota by poté mohla být zasílána data na displej. Při rozšíření robota o mechanický prvek, např. pohyblivou videokameru nebo mechanickou ruku, by bylo vhodné přeprogramovat joystick pro rotaci robota pro ovládání těchto mechanických prvků. Po dalších úpravách zdrojového kódu by bylo možné použít toto dálkové ovládání i pro létající roboty.

Všechny zdrojové kódy jsou nahrány na CD a přiloženy k této práci.

Literatura

[1] DI JASIO, Lucio: *Programming 32-bit microcontrollers in C: exploring the PIC32*. Second edition. Burlington, MA: Newnes, c2008, xxiii, 527 p. [cit. 2014-10-04]. ISBN 978-075-0687-096.

[2] HARRIS, David Money a Sarah L. HARRIS: *Digital Design and Computer Architecture*. Second edition. xxiv, 690 p. [cit. 2014-10-04]. ISBN 978-0-12-394424-5.

[3] Herout, Pavel: *Učebnice jazyka C, 1.díl.*Koop, 2002. ISBN 80-85828-21-9.

[4] Herout, Pavel: *Učebnice jazyka C, 2.díl.*Koop, 2002. ISBN 80-85828-50-2.

[5] chipKIT uC32: *chipKIT^{TM} uC32^{TM} Board Reference Manual*. [online] 9. 10. 2013
`http://ww1.microchip.com/downloads/en/DeviceDoc/chipKIT uC32_rm.pdf`

[6] PIC32 Peripheral Libraries for MPLAB C32 Compiler: *32-bit Peripheral Library Guide*. [online] 15. 10. 2013
`http://www.johnloomis.org/microchip/docs/32-bit-Peripheral-Library-Guide.pdf`

[7] PIC32MX3XX/4XX: *PIC32MX3XX/4XX Data Sheet*. [online] 15. 10. 2013
`http://ww1.microchip.com/downloads/en/DeviceDoc/61143H.pdf`

[8] SPFD5408B: *SPFD5408B Data Sheet*. [online] 25. 11. 2013
`http://pdf.datasheetarchive.com/indexerfiles/Datasheets-SW8/DSASW00146878.pdf`

[9] chipKIT Pmod Shield-Uno: *chipKIT^{TM} Pmod Shield-Uno Reference Manual*. [online] 2. 3. 2014
`www.digilentinc.com/...PMOD-SHIELD-UNO/chipKIT%20Pmod%20Shield-Uno_rm.pdf`

[10] PmodRF2: *PmodRF2^{TM} Reference Manual*. [online] 2. 3. 2014
`http://www.digilentinc.com/Data/Products/PMOD-RF2/PmodRF2_rm.pdf`

[11] MRF24J40: *MRF24J40 Data Sheet*. [online] 2. 3. 2014
`http://ww1.microchip.com/downloads/en/DeviceDoc/39776C.pdf`

[12] Microchip PIC32 | Mouser: *Mouser Electronics*. [online] 15. 3. 2014
`http://cz.mouser.com/new/microchip/microchip_pic32/`

[13] ASCII library: *ASCII.c*. [online] 27. 1. 2014
`https://github.com/vasyaod/STM32VGATextTerminal/blob/master/src/AsciiLib.c`

[14] ASCII library: *ASCII.h*. [online] 27. 1. 2014
`https://github.com/vasyaod/STM32VGATextTerminal/blob/master/src/AsciiLib.h`